华中科技大学

实验报告

课程名称：模拟电路与数字系统（三）实验

实验项目名称：VGA（2048）游戏

指导教师：左冬红

专业/班级：提高2101班

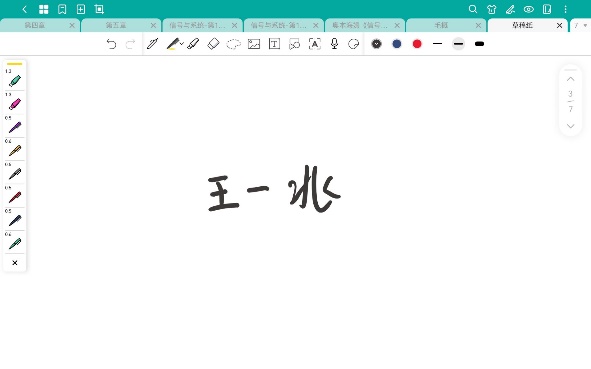
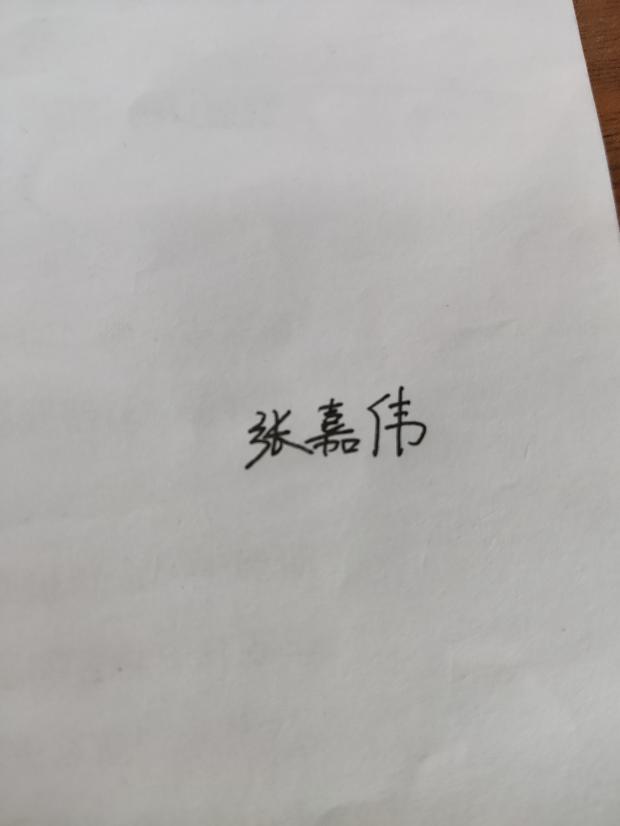
主要分工：

杨筠松（U202115980）：设计代码并完成主函数与逻辑运行模块main.c、logic.c、logic.h

王一兆（U202113912）：完成硬件平台搭建、图像数组生成image.c、image.h与PPT制作

张嘉伟（U202111218）：完成图像显示模块display.c、display.h与实验报告撰写

学生姓名（手写体电子签名）：





成绩/等级(批阅教师)：

实验执行日期： 2023 年5 月 3日

**实验内容 VGA（2048）游戏**

**一、 实验任务**

利用VGA显示器显示制作的2048小游戏

**二、 实验目的**

1. 学习VGA显示器的工作原理

2. 学会根据需求自主设计搭建硬件模块完成软件设计。

**三、 实验环境**

1. Windows 7 以上操作系统

2. 编辑工具：Vivado 、Vitis、Image2LCD

3.使用工具：VGA转VGA线、VGA显示屏

1. **实验原理**

**（1）VGA显示器原理**

VGA接口的基本原理是通过模拟信号来传输图像数据。图像数据由水平同步信号（HSYNC）和垂直同步信号（VSYNC）进行同步，同时使用红色（R）、绿色（G）和蓝色（B）三个颜色信号来表示像素的颜色。VGA接口的标准分辨率为640x480像素，每个像素可以表示16种颜色。通过调整信号的频率和时序，可以实现不同分辨率和颜色深度的图像显示。

**（2）VGA显示控制器TFT IP核的配置**

VGA显示控制器TFT IP核内部寄存器包括AR寄存器（显存高位地址），CR寄存器（显示属性控制），IESR寄存器（中断状态控制），偏移地址如下图所示，这便要求我们需要在软件中填入相应的值到对应地址。

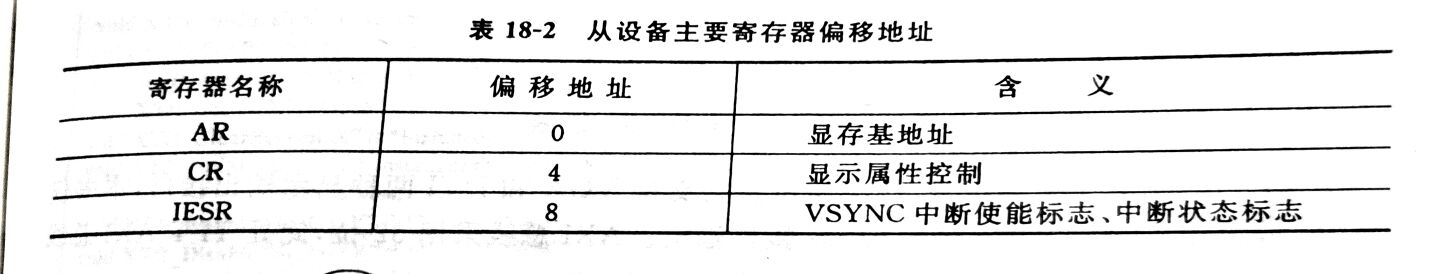


图1 从设备主要寄存器的偏移地址

TFT IP核默认的显存容量为2MB，从而AR寄存器仅存储了显存地址高11位，低21位为0，并且默认值为0x00000000;CR寄存器控制显示寄存器的扫描模式和显示、关闭状态，其默认寄存器的值为0x01。

**（3）硬件电路的搭建**

根据需求，搭建的硬件平台如下图所示：

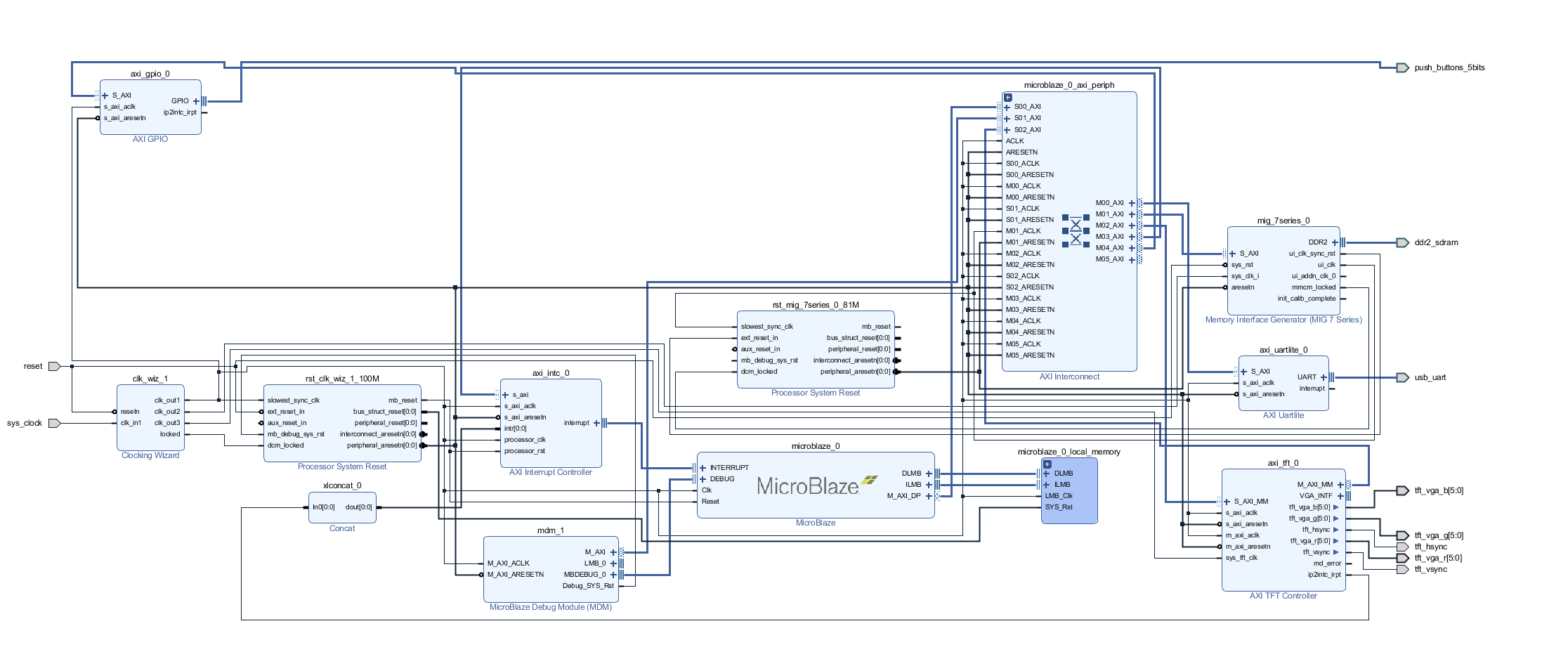


图2 硬件平台电路设计

为了实现利用外部按键进行控制，加入了GPIO的IP核。硬件平台搭建过程中，需要将VGA输出信号tft\_vga\_r[5:0]、tft\_vga\_b[5:0]、tft\_vga\_g[5:0]、tft\_hsync、tft\_vsync引出到外部，并且按照如下代码配置约束文件。

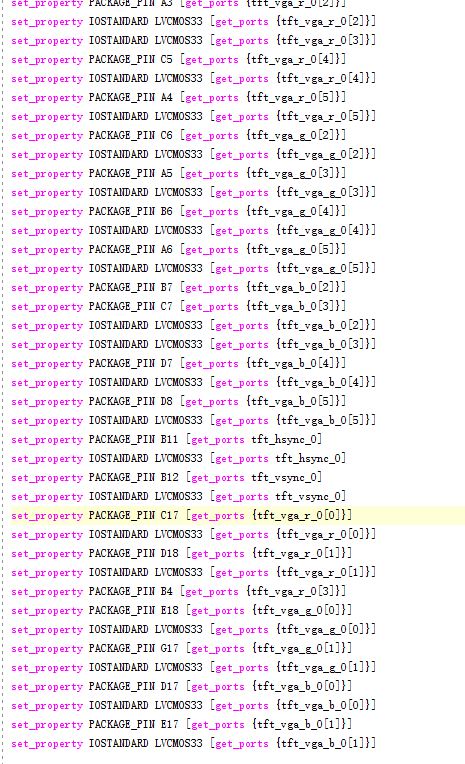
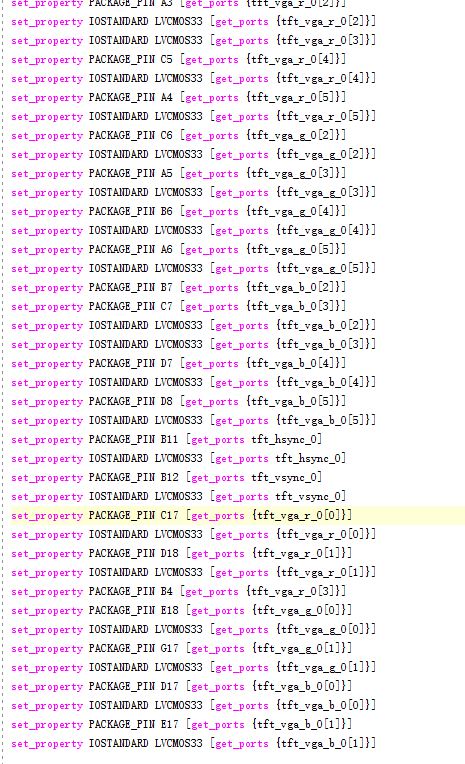


图3 TFT IP核约束文件

**（3）软件代码设计思路**

为了保证代码的独立性并便于分工，代码采用多文件联编来完成模块化设计，分别设计了main.c、display.c、display.h、image.c、image.h、logic.c、logic.h文件，其中main.c提供主函数入口供处理器调用，display模块主要将需要的图像展示到想要的位置，image模块完成了想要展示图像的存储，logic模块完成了2048游戏主体的代码逻辑。其中由于内存容量的问题，需要配置对应的脚本文件，具体按照书操作即可。为简化具体代码逻辑而突出代码功能描述，我们可以作出如下的流程图来展现工作流程（具体代码详见附录）：

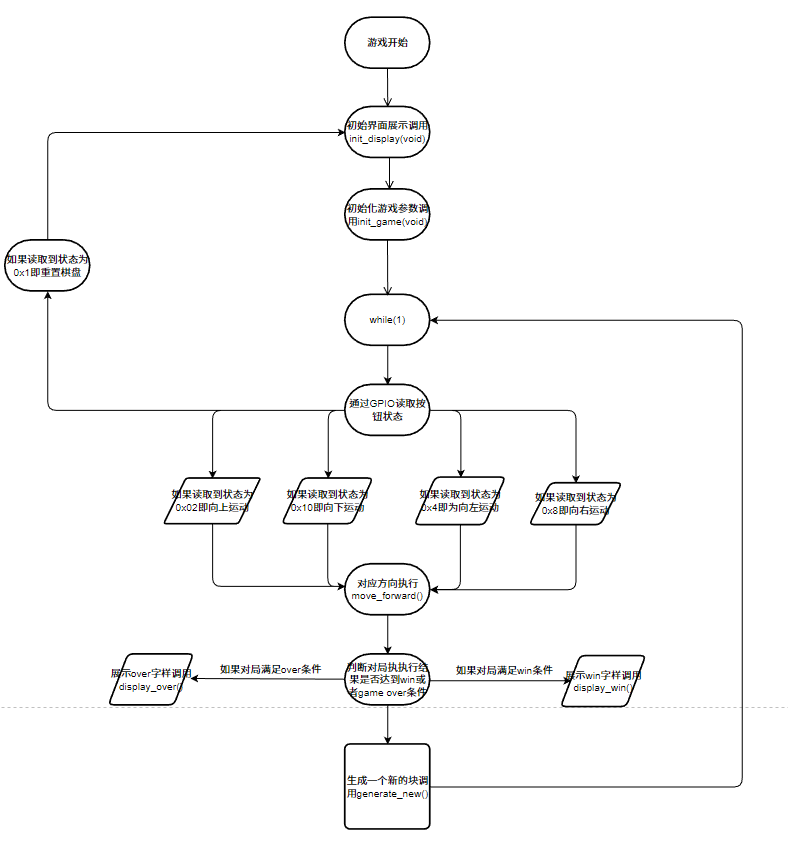


图4 程序运行流程图

1. **运行结果**

搭建具体的实验平台后，连接FPGA到VGA显示器，并且连接FPGA到电脑主机上，完成比特流生成后，烧录到FPGA开发板上，运行工程即可。

运行结果如下，其中：当场上不存在空闲的方格且无法移动时，显示“GAME OVER”；当场上出现2048时，显示“YOU WIN”（为了方便演示，图6中的设置是当出现256时，游戏获胜）

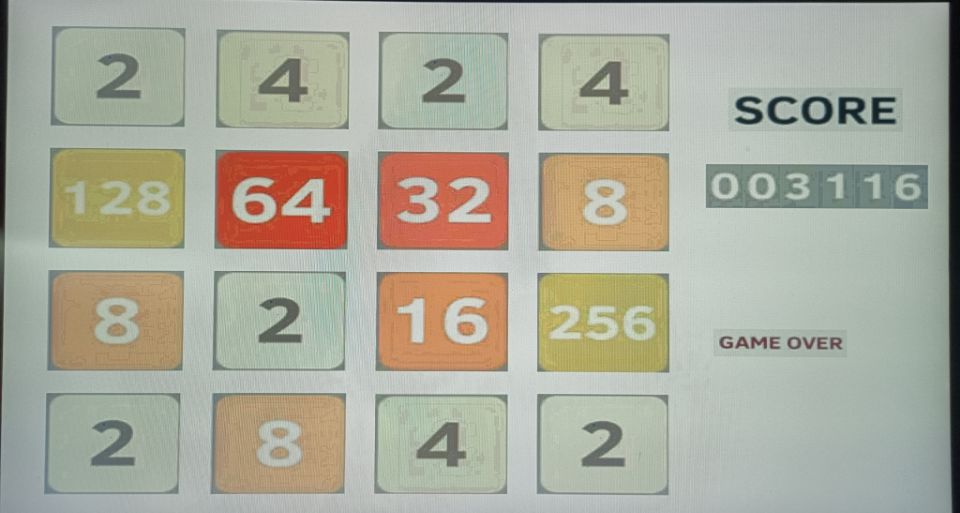


图5 “GAME OVER”显示

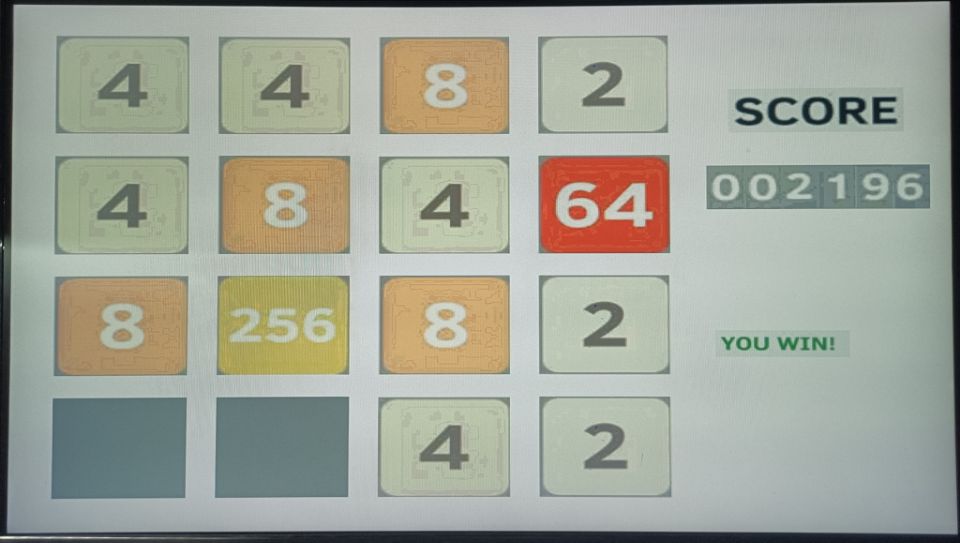


图6 “YOU WIN”显示

**六、 遇到的问题与解决办法**

1. **VGA无法正常显示**

解决办法：最开始我们使用的是VGA转HDMI线，希望可以接入个人显示器中，但是经过查阅资料并且多方验证后了解到DDR开发板无法驱动HDMI的显示功能，只好转换成VGA转VGA线，显示功能问题随即解决。

**2. 逻辑模块设计中滑动后碰撞无法消除滑块**

解决办法：考虑使用数据结构课程上面学到的知识进行处理，记录可以触碰的上一个模块的位置，分情况讨论判断是否上次已经碰撞完毕，是否为首个有值区块和是否可以完成碰撞。

**七、 实验总结**

本次实验是第一次自己设计需求，自己根据需求搭建硬件平台。在经过多次考虑之后，选择了VGA游戏并最终敲定了2048这个小游戏，通过五个案按键进行控制。起初的难点在于搭建硬件电路，我们最后只使用了一个GPIO核便完成了设计。为了方便工作的进行，我们在设计完硬件平台后，每个人进行分工完成软件的不同部分，采用多文件联编来完成模块化设计。这是一次比较成功的尝试，将学过的知识运用也给我们带来了很多收获。

**八、 附录**

（1）Display.c代码如下所示：

**#include** "display.h"

**#include** "image.h"

**#include** "xil\_io.h"

**extern** **unsigned** **int** maps[4][4];

**int** **check\_pos**(**int** val){

**int** s = 2;

**for** (**int** i = 1; i < 12; i++){

**if** (s == val) **return** i;

**else** s = s << 1;

}

**return** 0;

}

**void** **init\_display**(**void**) {

Xil\_Out32(TFT\_0\_BASEADDR + TFT\_AR\_OFFSET, TFT\_FRAME\_ADDR0);

**int** i, j;

**for** (i = 0; i < 480; i++)

**for** (j = 0; j < 640; j++)

Xil\_Out32(TFT\_FRAME\_ADDR0 + (4 \* (i \* 1024 + j)), 0xffffff); // clear the screen.

**for** (i = 0; i < 16; i++) display\_bulk(i, 0); // plot each grids for image

**for** (i = 80; i < 120; i++)

**for** (j = 500; j < 618; j++){

u32 color = Xil\_In32((**unsigned** **int**)gImage\_score + ((i - 80) \* 118 + (j - 500)) \* 4);

Xil\_Out32(TFT\_FRAME\_ADDR0 + 4 \* (1024 \* i + j - 10), color);

}

display\_score(0);

}

**void** **display\_bulk**(**int** pos, **int** val) {

**int** pos\_row\_start = 30 + (**int**)(pos / 4) \* 110, pos\_col\_start = 30 + (pos % 4) \* 110;

**int** i, j;

**for** (i = 0; i < 90; i++)

**for** (j = 0; j < 90; j++){

**int** pos\_row = pos\_row\_start + i, pos\_col = pos\_col\_start + j;

**int** addr = (pos\_row \* 1024 + pos\_col) \* 4;

**int** color = 0x808080;

**if** (val != 0) {

**int** color\_addr = (**unsigned** **int**)gImage\_number + 32400 \* (check\_pos(val) - 1) + ((i \* 90 + j) \* 4);

color = Xil\_In32(color\_addr);

}

Xil\_Out32(TFT\_FRAME\_ADDR0 + addr, color);

}

**return**;

}

**void** **display\_score**(**int** val){

**int** digits[6] = {0, 0, 0, 0, 0, 0};

**for** (**int** i = 5; i >= 0; i--){

digits[i] = val % 10;

val /= 10;

}

**for** (**int** i = 0; i < 6; i++){

**int** pos\_row\_start = 150 , pos\_col\_start = 5 + i \* 25;

**for** (**int** row = 0; row < 40; row++)

**for** (**int** col = 0; col < 25; ++col){

u32 color = Xil\_In32((**unsigned** **int**)gImage\_digits + 4000 \* digits[i] + ((row \* 25 + col) \* 4));

Xil\_Out32(TFT\_FRAME\_ADDR0 + 4 \* (1024 \* (row + pos\_row\_start) + (col + 470 + pos\_col\_start)), color);

}

}

}

**void** **display\_outcomes**(){

**for** (**int** i = 0; i < 16; i++) display\_bulk(i, maps[i / 4][i % 4]);

}

**void** **display\_message**(**int** occassion) {

**if** (occassion == 0) {

**for** (**int** row = 300; row < 325; row++)

**for** (**int** col = 480; col < 480 + 90; col++) {

u32 color = Xil\_In32((**unsigned** **int**)gImage\_over + (90 \* (row - 300) + (col - 480)) \* 4);

Xil\_Out32(TFT\_FRAME\_ADDR0 + 4 \* (row \* 1024 + col), color);

}

} **else** {

**for** (**int** row = 0; row < 24; row++)

**for** (**int** col = 0; col < 90; col++) {

u32 color = Xil\_In32((**unsigned** **int**)gImage\_win + ((90 \* row + col) \* 4));

Xil\_Out32(TFT\_FRAME\_ADDR0 + 4 \* ((row + 300) \* 1024 + col + 480), color);

}

}

}

（2）Logic.c代码如下所示：  
**#include** "logic.h"

**#include** "stdio.h"

**#include** "stdlib.h"

**#include** "display.h"

**extern** **unsigned** **int** maps[4][4];

**int** total\_score = 0;

**int** directions[4][2] = {{0, 1}, {0, -1}, {-1, 0}, {1, 0}};

**void** **init\_game**(**void**) {

**for** (**int** i = 0; i < 4; i++)

**for** (**int** j = 0; j < 4; j++)

maps[i][j] = 0;

total\_score = 0;

generate\_new();

}

**void** **generate\_new**(){

**int** pos\_empty[16], cnt = 0;

**for** (**int** i = 0; i < 16; i++)

**if** (maps[i / 4][i % 4] == 0) {

pos\_empty[cnt] = i;

cnt++;

}

**srand**(**rand**());

xil\_printf("random :%d\n", cnt);

**int** select\_pos = pos\_empty[**rand**() % cnt];

maps[select\_pos / 4][select\_pos % 4] = 2;

xil\_printf("random :%d %d\n", select\_pos / 4, select\_pos % 4);

display\_outcomes();

}

**void** **move\_forward**(**int** id){

**if** (id == 1) {

**for** (**int** col = 0; col < 4; col++) {

**int** ans[4] = {0}, cnt = 0, lst = -1;

**for** (**int** i = 0; i < 4; i++) {

**if** (maps[i][col] == 0) **continue**;

**if** (lst == -1) {

ans[cnt++] = maps[i][col];

lst = i;

} **else** **if** (maps[i][col] == maps[lst][col]){

ans[cnt - 1] \*= 2;

total\_score += ans[cnt - 1];

lst = -1;

} **else** {

ans[cnt++] = maps[i][col];

lst = i;

}

}

**for** (**int** i = 0; i < 4; i++)

maps[i][col] = ans[i];

}

display\_score(total\_score);

generate\_new();

} **else** **if** (id == 2) {

**for** (**int** col = 0; col < 4; col++) {

**int** ans[4] = {0}, cnt = 3, lst = -1;

**for** (**int** i = 3; i >= 0; i--) {

**if** (maps[i][col] == 0) **continue**;

**if** (lst == -1) {

ans[cnt--] = maps[i][col];

lst = i;

} **else** **if** (maps[i][col] == maps[lst][col]){

ans[cnt + 1] \*= 2;

total\_score += ans[cnt + 1];

lst = -1;

} **else** {

ans[cnt--] = maps[i][col];

lst = i;

}

}

**for** (**int** i = 3; i >= 0; i--)

maps[i][col] = ans[i];

}

display\_score(total\_score);

generate\_new();

} **else** **if** (id == 3){

**for** (**int** row = 0; row < 4; row++) {

**int** ans[4] = {0}, cnt = 0, lst = -1;

**for** (**int** i = 0; i < 4; i++) {

**if** (maps[row][i] == 0) **continue**;

**if** (lst == -1) {

ans[cnt++] = maps[row][i];

lst = i;

} **else** **if** (maps[row][i] == maps[row][lst]){

ans[cnt - 1] \*= 2;

total\_score += ans[cnt - 1];

lst = -1;

} **else** {

ans[cnt++] = maps[row][i];

lst = i;

}

}

**for** (**int** i = 0; i < 4; i++)

maps[row][i] = ans[i];

}

display\_score(total\_score);

generate\_new();

} **else** **if** (id == 4) {

**for** (**int** row = 0; row < 4; row++) {

**int** ans[4] = {0}, cnt = 3, lst = -1;

**for** (**int** i = 3; i >= 0; i--) {

**if** (maps[row][i] == 0) **continue**;

**if** (lst == -1) {

ans[cnt--] = maps[row][i];

lst = i;

} **else** **if** (maps[row][i] == maps[row][lst]){

ans[cnt + 1] \*= 2;

total\_score += ans[cnt + 1];

lst = -1;

} **else** {

ans[cnt--] = maps[row][i];

lst = i;

}

}

**for** (**int** i = 3; i >= 0; i--)

maps[row][i] = ans[i];

}

display\_score(total\_score);

generate\_new();

}

**if** (check\_lose()) display\_message(0);

**if** (check\_win()) display\_message(1);

}

**int** **check\_lose**(**void**){

// for (int col = 0; col < 4; col++) if (maps[0][col] == 0) return 0;

**for** (**int** i = 0; i < 4; i++)

**for** (**int** j = 0; j < 4; j++){

**if** (maps[i][j]) {

**for** (**int** k = 0; k < 4; k++){

**int** new\_x = i + directions[k][0], new\_y = j + directions[k][1];

**if** (new\_x < 0 || new\_x > 3 || new\_y < 0 || new\_y > 3) **continue**;

**if** (maps[new\_x][new\_y] == maps[i][j]) **return** 0;

}

} **else** **return** 0;

}

**return** 1;

}

**int** **check\_win**(**void**) {

/\*\*

\* check whether the maps are full or has encountered 2048

\* if the answer is yes, return 1

\* else return 0.

\*/

**for** (**int** i = 0; i < 4; i++)

**for** (**int** j = 0; j < 4; j++){

**if** (maps[i][j] == 256) **return** 1;

}

**return** 0;

}

（3）Main.c的代码如下所示：

**#include** "xintc\_l.h"

**#include** "xgpio\_l.h"

**#include** "xparameters.h"

**#include** "xio.h"

**#include** "xil\_exception.h"

**#include** "display.h"

**#include** "logic.h"

**#define** TFT\_BASEADDR 0x44a00000

**#define** TFT\_FRAME\_BASEADDR 0x80000000

**unsigned** **int** maps[4][4] = {0};

**int** total\_scores = 0;

**int** **main**(){

init\_display();

init\_game();

Xil\_Out32(XPAR\_AXI\_GPIO\_0\_BASEADDR + XGPIO\_TRI\_OFFSET, 0x1f);

**unsigned** **short** last\_status = 0;

**while** (1) {

// loop for reading the statues from the buttons.

**for**(**int** i = 0; i < 100000; i++);

**unsigned** **short** status = 0;

status = Xil\_In16(XPAR\_AXI\_GPIO\_0\_BASEADDR + XGPIO\_DATA\_OFFSET) & 0x1f;

**if** (status != last\_status) {

last\_status = status;

} **else** **continue**;

**int** direction = 0;

**switch** (status) {

**case** 0x2:

direction = 1;

**break**;

**case** 0x10:

direction = 2;

**break**;

**case** 0x4:

direction = 3;

**break**;

**case** 0x8:

direction = 4;

**break**;

**case** 0x1:

init\_display();

init\_game();

**break**;

**default**:

**continue**;

}

move\_forward(direction);

}

**return** 0;

}